Amendments to the Specification:

Please amend the specification as follows:

Please amend the paragraph bridging pages 1 and 2 (page 1, line 24 to page 2, line 14). with the following rewritten paragraph:

In a real-time control system, an event that is an impulse (stimulation) from ~~an~~ outside or ~~an~~ inside of the real-time control system, for example, various signals and a state that is a behavior of the real-time control system, for example, waiting to receive various signals, are combined ~~complicatedly~~ with difficulty. Also, there are many processes corresponding to these combinations, namely, many actions that are processes executed by the real-time control system when a specific event occurs under a specific state. As one ~~of techniques~~ technique for developing a program to be installed in such a real-time control system, there is a program development method using a state-transition matrix. The state-transition matrix is shown by a two-dimensional matrix in which an event and a state are respectively arranged in a row or a column and an action corresponding to an intersection point (cell) of the event and the state and a transition destination after the action are arranged. According to this program development method, though the real-time control system becomes large and becomes complicated, it is possible for an inexperienced person in basic design to execute the basic design and it is also possible to save labor and to shorten a development period.

Please amend the first full paragraph on page 20, lines 3-16, with the following rewritten paragraph:

The man-machine interface 1 is provided with a display section 1a, a mouse 1b, a keyboard 1c and [[a]] the like. A user operates the mouse 1b and the keyboard 1c so as to input information (such as a state, an event, an action and a transition destination) necessary to generate a state-transition matrix. Also, in order to execute an emulation for each event based on a state-transition

matrix of a real-time control system designed using the state-transition matrix by the in-circuit emulator 14, a cursor is moved by the cursor key of the mouse 1b or the keyboard 1c in an event display area displayed in the display section 1a and a cursor key of the mouse 1b or the keyboard 1c is pushed, and thereby the man-machine interface 1 is used to indicate an input of the event and an emulated result supplied from the debugger 13 is displayed on the display section 1a.

Please amend the paragraph bridging pages 20 and 21 (page 20, line 17 to page 21, line 16), with the following rewritten paragraph:

The state-transition matrix editor 2 generates and edits a state-transition matrix based on the state, the event, the action, the transition destination or [[a]] the like input via the man-machine- interface 1, and ~~memorizes~~ stores information concerning the state-transition matrix in the state-transition matrix memory section 4. The event pseudo-generating editor 3~~, in order to pseudo-generate~~ generates a same event as the event generated based on a signal corresponding to ~~the~~ an operation supplied from the target system by operating various keys, switches or [[a]] the like of the target system or based on various data or signals transmitted from elements of another CPU or a semiconductor device[[,]] The event pseudo-generating editor 3 does this by detecting that the cursor is moved, or that the cursor key of the mouse 1b or the keyboard 1c is in the display area of the event in the state-transition matrix displayed on the display section 1a and a left button of the mouse 1b is clicked or a return key is pushed~~, information~~. Information concerning the state-transition matrix ~~memorized~~ stored in the state-transition matrix memory section 4 is referred based on information (such as each event generating technique) input via the man-machine interface 1 necessary for pseudo-generating input, and event pseudo-generating information, which is information concerning [[to]] an event to be pseudo-generated, is generated and is ~~memorized~~ stored in the event pseudo-generating information memory section 5. Both [[of]] the state-transition

matrix memory section 4 and the event pseudo-generating information memory section 5 are storage media having large memory capacities, for example, a semiconductor memory such as a RAM, a FD (a Floppy Disc) and a HD (a Hard Disk), and information of the state-transition matrix and the event pseudo-generating information are respectively stored in the storage media.

Please amend the paragraph bridging pages 21 and 22 (page 21, line 17 to page 22, line 16), with the following rewritten paragraph:

The generator 6, based on information concerning the state-transition matrix read from the state-transition matrix memory section 4 and the event pseudo-generating information read from the event pseudo-generating information memory section 5, automatically generates a program (source program) described in a programming language, for example, in C language or [[a]] the like to be installed in the target system 16 and ~~memorized~~ stored in the program memory section 7. The generator 6, also based on the event pseudo-generating information read from the event pseudo-generating information memory section 5, automatically generates an event pseudo-generating routine described in a programming language similar to the source program and ~~memorizes~~ stores the event pseudo-generating routine in the event pseudo-generating routine memory section 8. The source program includes a main routine for executing main processes of the target system 16 and an event normal generating routine, based on a signal supplied from the target system 16 by operating various keys, various switches or [[a]] the like forming the target system 16 and corresponding to the operation and based on various data and various signals transmitted from elements such as another CPU and a semiconductor device, for detecting which key or which switch is operated or which element transmits data and a signal and notifying the main routine of that, namely, for normally generating an event. Each of the program memory section 7 and the event pseudo-generating routine memory section 8 is a semiconductor memory such as a RAM or a storage medium of a large memory capacity such

as a FD or a HD. The program memory section 7 and the event pseudo-generating routine memory section 8 respectively ~~memorize~~ store the source program and the event pseudo-generating routine.

Please amend the paragraph bridging pages 22 and 23 (page 22, line 17 to page 23, line 2), with the following rewritten paragraph:

The compiler 9 converts the source program read from the program memory section 7 ~~in to~~ into an object program described in a machine language executable by the CPU in the target system 16 and ~~memorize~~ stores the object program in the machine language code memory section 10. The compiler 9 also converts the event pseudo-generating routine read from the event pseudo-generating routine memory section 8 and described in a programming language into the event pseudo-generating routine described in a machine language similar to the object program, and ~~memorizes~~ stores the event pseudo-generating routine in the machine language code memory section 10. The machine language code memory section 10 is a semiconductor memory such as a RAM or a storage medium of a large memory capacity such as a FD or a HD, and ~~memorizes~~ it stores the object program and the event pseudo-generating routine described in the machine language.

Please amend the first full paragraph on page 23, lines 3-10, with the following rewritten paragraph:

...put section 11 detects a position of a cursor when an operator moves the cursor to a display area such as an event and a state in the state-transition matrix displayed on the display section 1a and clicks the left button of the mouse 1b or pushes the return key, of the keyboard 1c and supplies position information in this case to the analysis section 12. In other words, the input section 11 in the first embodiment functions as a position detecting section for an event, a state or [[a]] the like.

Please amend the second full paragraph on page 23, lines 11-22, with the following rewritten paragraph:

The analysis section 12 converts the position information supplied from the input section 11 into an event code, a state code or [[a]] the like. The analysis section 12 also controls the debugger 13 so as to set a state corresponding to the state code as an initial state for starting emulation of the object program in the in-circuit emulator 14 and refers to the event pseudo-generating information read from the event pseudo-generating information memory section 5 based on the event code and controls the debugger 13 and re-writes information used by the event pseudo-generating routine ~~memorized~~ stored in a predetermined memory area of a RAM 23 (Fig. 2) in the in-circuit emulator 14, and thereby pseudo-generates an event corresponding to the event code.

Please amend the second full paragraph on page 24, lines 8-21, with the following rewritten paragraph:

The in-circuit emulator 14 is mainly provided with the evaluation chip 21, a ROM 22, the RAM 23 and an I/O port 24. The evaluation chip 21, for the object program test, is provided with a circuit for debugging in addition to a CPU core for processing a program, a terminal for only debugging in addition to terminals of a real chip and has a function equal to the real chip. The ROM 22 ~~memorizes~~ stores the object program ~~memorized~~ stored in the machine language code memory section 10 and the event pseudo-generating routine described in the machine language. The RAM 23 is used while the evaluation chip 21 executes the object program and a part of contents of the RAM 23 is re-written by the debugger 13. The I/O port 24 is connected to the target system 16 via the emulation probe 15, and various signals transmitted between the evaluation chip 21 and the target system 16 are input and output.

Please amend the paragraph bridging pages 23 and 24 (page 23, line 22 to page 24, line 3), with the following rewritten paragraph:

The target system 16 is an example of a CD player and is mainly provided with a housing 31, a keyboard 32, a display device 33, a mechanism 34, a tray 35, an analog signal processing circuit 36, a digital signal processing circuit 37, a RAM 38, a D/A converter 39, an amplifier 40, a speaker 41 and driver 42 to driver 46. The target system 16 is connected with the in-circuit emulator 14 via the emulation probe 15 by engaging the housing 31 with a CPU socket provided at a top of the emulation probe 15. The keyboard 32 is provided with various switches and keys such as a power switch, a play key and a stop key. The display device 33 displays a passing time, a track number and [[a]] the like of music.

Please amend the first full paragraph on page 25, lines 4-14, with the following rewritten paragraph:

The mechanism 34 is mainly provided with an optical pickup 47 for reading digital information ~~memorized~~ stored by pitting on a signal record surface of a CD using a laser beam, a spindle motor 48 for rotationally driving the CD 55 at a constant linear velocity, a tray opening and shutting motor 49 controlled by the evaluation chip 21 via the I/O port 24, the emulation probe 15, the housing 31 and the driver 45 and for opening and shutting the tray 35 and a field motor 50 controlled by the evaluation chip 21 via the I/O port 24, the emulation probe 15, the housing 31 and the driver 46 and for moving the optical pickup 47 in a radius direction of the CD 55.

Please amend the first full paragraph on page 26, lines 14-21, with the following rewritten paragraph:

Next, description will be given with respect to an operation of the program development apparatus according to the first embodiment as described above. First, the target system 16 as a CD player is provided with various functions for reproducing music or [[a]] the like based on the digital information recorded in the signal record surface of the CD 55 using pits. In the first embodiment, it is assumed that a serial processing is tested in the object programs as described later.

Please amend the paragraph bridging pages 27 and 28 (page 27, line 22 to page 28, line 4), with the following rewritten paragraph:

The operator operates the mouse 1b and the keyboard 1c by referring to display screen of the display section 1a constituting the man-machine interface 1, and inputs information (such as a state, an event, an action and a transition point) required to generate a state-transition matrix shown in Fig. 3 based on the operation of the target system 16. With this operation, the state-transition matrix editor 2 generates the state-transition matrix shown in Fig. 3, displays the state-transition matrix on the display section 1a constituting the man-machine interface 1, and ~~memorizes~~ stores information concerning the state-transition matrix in a predetermined memory area in the state-transition matrix memory section 4.

Please amend the paragraph bridging pages 28 and 29 (page 28, line 15 to page 29, line 4), with the following rewritten paragraph:

In a highest row in Fig. 3, "stopping" indicates a state (hereafter, called "state 1") in that the spindle motor 48, the tray opening and shutting motor 49 and the field motor 50 are stopped and the tray 35 is drawn into the CD player or the opening and shutting motor 49 is driven clockwise, "tray opening" indicates a state (hereafter, called "state 2") in that the spindle motor 48, the tray opening and shutting motor 49 and the field motor 50 are stopped and the tray 35 is drawn out from the CD player or the opening and shutting motor 49 is

driven counterclockwise, "TOC reading" indicates a state (hereafter, called "state 3") in that the tray 35 is drawn into the CD player, the tray opening and shutting motor 49 is ~~stooped~~ stopped, both of the spindle motor 48 and the field motor 50 are driven and the TOC receded in the most internal circumference of the CD 55 is read, "time record display" indicates a state (hereafter, called "state 4") in that the tray 35 is drawn into the CD player, all of the spindle motor 48, the tray opening and shutting motor 49 and the field motor 50 are stopped and time records of all pieces of music are displayed on the display device 33.

Please amend the second full paragraph on page 32, lines 15-20, with the following rewritten paragraph:

In a cell (2, 7), "×" indicates that any action is executed ~~is executed~~ in accordance with occurrences of the "state 2" and the "state 7" when the object program is completed or no action is executed and no state is transited at a current step. A meaning of "×" is similar in other cells, and therefore, no explanation thereof will be given.

Please amend the first full paragraph on page 35, lines 7-25, with the following rewritten paragraph:

Next, the operator instructs an input of the event not by actually operating the tray key and the search key forming the keyboard 32 of the target system 16 but by moving the cursor on the display area of the state-transition matrix displayed on the display section 1a with the cursor key of the mouse 1b and the keyboard 1c so as to pseudo-generate the event and, in order to execute an emulation for each event with the in-circuit emulator 14, operates the mouse 1b and the keyboard 1c while referring to the state-transition matrix (Fig. 3) displayed on the display section 1a constituting the man-machine interface 1 and inputs information necessary to pseudo-generate each event (a technique of generating the event in accordance with [[a]] an actual kind of ~~actually~~ event, ~~an~~

for each event or [[a]] the like). Concerning kinds of events, there is an in-mail type in that an internal event generated in a cell of a state-transition matrix is notified to another state-transition matrix when [[a]] state-transition matrixes are layered, a function-call type in that a function for executing a group of processes is called in addition to the message-type, the flag-type event and the interrupt-type which are described above.

Please amend the paragraph bridging pages 35 and 36 (page 35, line 26 to page 36, line 6), with the following rewritten paragraph:

With this operation, the event pseudo-generating editor 3, based on the information necessary for pseudo generation input via the man-machine interface 1 (the technique of generating the event in accordance with each event, refers to information relative to the state-transition matrix ~~memorized~~ stored in the state-transition matrix memory section 4 and generates event pseudo-generating information which is information relative to an event to be pseudo-generated so as to ~~memorized~~ store the event pseudo-generating information in the event pseudo-generating information memory section 5.

Please amend the first full paragraph on page 36, lines 7-24, with the following rewritten paragraph:

The generator 6 reads the information relative to the state-transition matrix from the state-transition matrix memory section 4 and reads the event pseudo-generating information from the event pseudo-generating information memory section 5, automatically generates a source program described in C language or [[a]] the like based on ~~these~~ this information and ~~memorizes~~ stores the source program in the program memory section 7, and automatically generates an event pseudo-generating routine for pseudo-generating an event and ~~memorizes~~ stores the event pseudo-generating routine in the event pseudo-generating routine memory section 8. The source program includes a main routine for executing main processes of the target system 16 and an event

normal generating routine for notifying the main routine which key or switch is operated based on a signal supplied from the target system 16 by operating a key, a switch or a like in the target system 16 in accordance with the operation or based on various data and signals transmitted from another element such as a CPU or a semiconductor device, namely, for generating an event normally.

Please amend the first full paragraph on page 38, lines 8-14, with the following rewritten paragraph:

As described above, when the source program and the event pseudo-generating routine is described in C language and are ~~memorized~~ stored in the program memory section 7 and the event pseudo-generating routine memory section 8, the compiler 9 reads the source program described in C language from the program memory section 7, compiles the source program into an object program and ~~memorizes~~ stores it in the machine language code memory section 10.

Please amend the second full paragraph on page 38, lines 15-24, with the following rewritten paragraph:

Also, the compiler 9 reads the event pseudo-generating routine described in C language from the event pseudo-generating routine memory section 8, compiles the read event pseudo-generating routine into an event pseudo-generating routine described in a same machine language as the object program and ~~memorizes~~ stores it in the machine language code memory section 10. The object program and the event pseudo-generating routine described in the machine language and ~~memorized~~ stored in the machine language code memory section 10 are written in a ROM by a ROM writer, and the ROM is installed in the in-circuit emulator 14 as the ROM 22.

Please amend the third full paragraph on page 39, lines 11-27, with the following rewritten paragraph:

Then, the operator moves the mouse 1b or the keyboard 1c to a display area of a state selected as an initial state of the emulation to be started (in this case, the "state 6" in that the tray 35 is drawn into the CD player, the tray opening and shutting motor 49 is stopped, both of the spindle motor 48 and the ~~filed~~ field motor 50 are driven and the top of the second piece of music is searched, "moving to second music" in Fig. 6) and clicks the left button or pushes the return key. With this operation, the input section 11 detects a position of the cursor in the display area of the state selected by the operator and supplies position information to the analysis section 12, and the analysis section 12 converts the position information supplied from the input section 11 into a state code corresponding to the position, namely, a code of the "state 6" in this case, and then controls the debugger 13 while setting a state corresponding to the state code (the state 6 in this case, "moving to second music" in Fig. 6 to the in-circuit emulator 14.

Please amend the paragraph bridging pages 39 and 40 (page 39, line 28 to page 40, line 8), with the following rewritten paragraph:

With this operation, the debugger 13 controls the in-circuit emulator 14 ~~starts~~ to start to execute a series of processes from the state set as the initial state, namely, in this case, from the "state 6" in that the tray 35 is drawn into the CD player, the tray opening and shutting motor 49 is stopped, both of the spindle motor 48 and the field motor 50 are driven and the top of the second piece of music is searched. In this case, it is assumed that the event pseudo-generating routine ~~memorized~~ stored in the ROM 22 together with the main routine and the event normal generating routine is periodically executed by a timer interruption or [[a]] the like.

Please amend the first full paragraph on page 40, lines 9-26, with the following rewritten paragraph:

Then, when the operator moves the cursor with the mouse 1b or the cursor key of the keyboard 1c to a display area of the event of input of a signal corresponding to the play key which is pushed ("play key input" in Fig. 6) and clicks the left button or pushes the return key, the input section 11 detects a position of the cursor in the display area of the "event 7" and supplies position information to the analysis section 12. Therefore, the analysis section 12 converts the position information supplied from the input section 11 into an event code of the "event 7" corresponding to the position and, based on the event code, while referring to event pseudo-generating information read from the event pseudo-generating information memory section 5, controls the debugger 13, rewrites information used by the event pseudo-generating routine ~~memorized~~ stored in a predetermined area of the RAM 23 in the in-circuit emulator 14 and rewrites a value of the variable FakeEvent into a same value of the constant EVENT_KEY_ PLAY which is a constant corresponding to the event code of the "event 7" in this case.

Please amend the second full paragraph on page 41, lines 18-21, with the following rewritten paragraph:

In addition, following operations of the debugger 13 and the in-circuit emulator 14 are approximately similar to those of the conventional debugger and the conventional in-circuit emulator, and therefore, explanations thereof will be omitted.

Please amend the paragraph bridging pages 41 and 42 (page 41, line 22 to page 42, line 5), with the following rewritten paragraph:

As described above, according to the first embodiment, only when the operator moves the cursor with the mouse 1b or the cursor key of the keyboard 1c to a display area of the desirable event of occurrence among the plural events in the state-transition matrix displayed at the left side of the emulation mode screen shown in Fig. 6 and clicks the left button or pushes the return key, the

event pseudo-generating routine pseudo-generates a same event as this event[[,]]. This occurs in spite of a process which is first executed by operating plural switches and plural keys in order plural times, and thus it is possible to test any process of the target system 16 in any order simply and in a short time without using hardware of the target system 16. As a result, it is possible to shorten a development period of the target system 16.

Please amend the first full paragraph on page 42, lines 6-10, with the following rewritten paragraph:

In addition, in Fig. 1 and Fig. 2, though the in-circuit emulator 14 is connected to the target system 16 via the emulation probe 15, this configuration is convenient for explanations or [[a]] the like of the state-transition matrix shown in Fig. 3, and therefore, it is unnecessary to prepare the target system 16.

Please amend the paragraph bridging pages 43 and 44 (page 43, line 18 to page 43, line 7), with the following rewritten paragraph:

The input section 61, in addition to a function of the input section 11 shown in Fig. 1, is provided with a function for ~~memorizing~~ storing an order of events selected with a mouse 1b or a keyboard 1c by an operator and an input timing of each event (hereafter, called an input event log) in an internal memory and then for supplying the input event log to the script generating section 63. The script generating section 63, based on the input log supplied from the input section 61, generates a script file and ~~memorizes~~ stores it in the script memory section 64. The script file is a file of a timing chart format, a text format or a message sequence chart format in which an order and an occurrence timing of generating each order, a timing of which elements in the target system 16 should be operated according to specifications or [[a]] the like is described in order to execute an emulation based on the state-transition matrix of a target

system 16 designed by the state-transition matrix. The script memory section 64 is a semiconductor memory such as a RAM or a storage medium of a large storage capacity, such as an FD or a HD, and ~~memorizes~~ stores the script file.

Please amend the first full paragraph on page 44, lines 8-28, with the following rewritten paragraph:

The script editor 65, based on an event which should be occurred so as to execute the emulation input using an man-machine interface 1, an occurrence timing of the event or an occurrence frequency, edits the script file read from the script memory section 64 and then ~~memorizes~~ stores it in the script memory section 64 again. The script analysis section 66 sequentially supplies position information of each event described in the script file read from the script memory section 64 in a corresponding display area of the state-transition matrix displayed on a display section 1a to the analysis section 62 in order and at a timing described in the script file. The analysis section 62, in addition to a function of the analysis section shown in Fig. 1, is provided with a function for converting position information supplied from the script analysis section 66 into an event code corresponding to the position and for referring to the event pseudo-generating information read from an event pseudo-generating information memory section 5 based on the event code, controlling the debugger 13, rewriting the information used by the event pseudo-generating routine ~~memorized~~ stored in the predetermined memory area of a RAM 23 (Fig. 2) in an in-circuit emulator 14 and thereby pseudo-generating an event corresponding to the event code.

Please amend the first full paragraph on page 45, lines 12-22, with the following rewritten paragraph:

With this operation, the input section 61 detects a position of the cursor in the display area of the "event 7" , supplies the position information to the

analysis section 62, temporarily ~~memorizes~~ stores the input event log which is a timing of the event 7 and an input timing in the internal memory, and then supplies the input event log to the script generating section 63. Therefore, the script generating section 63, based on the input event log which is the timing of the "event 7" and the input timing supplied from the input section 61, generates a script file for generating the "event 7" once and ~~memorizes~~ stores the script file into the script memory section 64.

Please amend the paragraph bridging pages 45 and 46 (page 45, line 23 to page 46, line 11), with the following rewritten paragraph:

Therefore, when the program development apparatus is made to execute the same emulation successively, the operator only makes the script analysis section 66 read the script file from the script memory section 64 and supplies the position information of the "event 7" described in the script file in a corresponding display area in the state-transition matrix displayed on the display section 1a at the occurrence timing described in the script file without moving the cursor to the display area of the "event 7" in the state-transition matrix displayed at the left side of the emulation mode screen again. As a result, the analysis section 62 converts the position information into the event code corresponding to the position and, based on the event code, refers to the event pseudo-generating information read from the event pseudo-generating information memory section 5, controls the debugger 13 and rewrites the information used by the event pseudo-generating routine ~~memorized~~ stored in the predetermined storage area of the RAM 23 in the in-circuit emulator 14, and thereby the "event 7" corresponding to the event code is pseudo-generated.

Please amend the paragraph bridging pages 46 and 47 (page 46, line 12 to page 47, line 11), with the following rewritten paragraph:

Also, when the program development apparatus is made to execute an emulation for generating the "event 7" one hundred times, the operator, first,

operates the mouse 1b or the keyboard 1c while referring to display on the display section 1a in the man-machine interface 1, makes the script editor 65 read the script file from the script memory section 64 so as to display the script file on the display section 1a, and changes an occurrence frequency from once to one hundred times. With this operation, the script editor 65 changes the occurrence frequency of the "event 7" of the script file read from the script memory section 64 from once to one hundred times and then ~~memorizes~~ stores the script file in the script memory section 64 again. Then, the operator makes the script analysis section 66 read a new script file from the script memory section 64 and instructs to supply the position information of the "event 7" described in the script file in a corresponding display area in the state-transition matrix displayed on the display section 1a at the occurrence timing and the occurrence frequency (one hundred times in this case) described in the script file. With this operation, the analysis section 62 converts the position information supplied from the script analysis section 66 into an event code corresponding to the event code and, based on the event code, refers to the event pseudo-generating information read from the event pseudo-generating information memory section 5, controls the debugger 13 and rewrites the information used by the event pseudo-generating routine memorized in the predetermined storage area of the RAM 23 in the in-circuit emulator 14 and thereby the "event 7" corresponding to the event code is pseudo-generated one hundred times.

Please amend the paragraph bridging pages 48 and 49 (page 48, line 6 to page 49, line 3), with the following rewritten paragraph:

For example, in the first embodiment and the second embodiment, the in-circuit emulator 14 is mainly provided with the evaluation chip 21, the ROM 22, the RAM 23 and the I/O port 24 and is connected with the target system 16 via the emulation probe 15 by engaging the housing 31 in the target system 16 with the CPU socket provided at the top of is engaged the emulation probe 15, however, the present invention is not limited to this. For example, in order to

cope with a high-speed CPU, an evaluation chip is provided on the emulation probe of which an end is connected to an in-circuit emulator, the emulation probe is engaged with a housing into which a real chip installed on a print substrate forming hardware of a target system is inserted, and thereby the in-circuit emulator and the hardware of the target system may be connected via the emulation probe. In this case, as the hardware of the target system, a ROM in which an object program to be tested is stored and a print substrate on which a RAM used in executing the object program is mounted are needed at least. Also, in order to test the object program in a near real state, not an evaluation chip but a chip may be used in which a circuit, a memory and a terminal at least needed for trace are provided on a CPU chip of an internal structure and a number of terminals similar those of the real chip. In this case, as the hardware of the target system, the CPU chip of the above-described structure, a ROM in which an object program to be tested is ~~memorized~~ stored and a print substrate on which a RAM used in executing the object program are needed at least, an in-circuit emulator is unnecessary and the CPU chip and the debugger may be connected via a probe.

Please amend the paragraph bridging pages 49 and 50 (page 49, line 14 to page 50, line 13), with the following rewritten paragraph:

Furthermore, in the first embodiment and the second embodiment, each section is represented by hardware, however, the present invention is not limited to this. In other words, the program development apparatus may be a computer including a CPU, internal storage units such as a ROM and a RAM, external storage units such as an FDD (Floppy Disk Driver), a HDD (Hard Disk Driver) and a CD-ROM driver, an input unit and an output unit. The state-transition matrix editor 2, the event pseudo-generating editor 3, the generator 6, the compiler 9, the input section 11 (61), the analysis section 12 (62), the debugger 13, the script generating section 63, the script editor 65, the script analysis section 66 and a like are carried out by the CPU, and these functions may be ~~memorized~~

stored in a semiconductor memory such as a ROM, a storage medium such as an FD, an HD and a CD-ROM as a program development program. In this case, the internal storage unit or the external storage unit becomes the state-transition matrix memory section 4, the event pseudo-generating information memory section 5, the program memory section 7, the event pseudo-generating routine memory section 8, the machine language code memory section 10, the script memory section 64 and [[a]] the like. The program development program is read from the storage medium into the CPU and controls the operation of the CPU. When the program development program is started, the CPU functions as the state-transition matrix editor 2, the event pseudo-generating editor 3, the generator 6, the compiler 9, the input section 11 (61), the analysis section 12 (62), the debugger 13, the script generating section 63, the script editor 65, the script analysis section 66 and [[a]] the like, and the above-described processes are executed by control of the program development program.